Supplementary Material

Abstract

This document provides some supplementary contents for the paper: A geometry-aware attention network for semantic segmentation of MLS point clouds. Section 1 of this document presents differences between our proposed method and existing methods. Section 2 details the mechanism of our designed GAAM from the feature level. More experimental results and analysis can be found in Section 3.

1 Comparison with existing methods

We further theoretically compare our proposed point network with other state-of-the-arts. We start by describing their differences on how to extract the local features from point clouds. As a pioneer work, PointNet (Qi *et al*. 2017a) processes input points individually and actually has no local feature extraction. The extended PointNet++ (Qi *et al*. 2017b) extracts local features by grouping local points in a fixed region scale. Instead of directly operating on local points, the subsequent graph-based methods such as DGCNN (Wang *et al*. 2019b) conduct local feature extraction of each point by attending over its neighbours in a graph-like region. Furthermore, various attention strategies, such as the graph attention used in GPM (Wang *et al*. 2019b), the self-attention used in PCT (Guo *et al*. 2021a) and GAPointNet (Chen *et al*. 2021), are applied to highlight different attention weights on the neighbourhood of each point to facilitate local feature learning. Encouraged by this, we develop a novel geometry-aware attention module to conduct the local feature extraction on MLS point clouds. Our designed GAAM adopts a multi-heads strategy and refines the self-attention in each single-head by constructing a local weighted graph for each input point and combining a reliable neighbouring embedding to efficiently capture local discriminative features of point cloud. It is worth noting that in the normal local graph constructed by DGCNN and GAPointNet, the contribution coefficient of neighbours to the central point is equal to 1, which may introduce irrelevant information when conducting the edge convolution. While in our local weighted graphs, the geometric similarity between the central point and its neighbours is significantly exploited to assign a proper contribution coefficient to each edge of the central point. The contribution coefficient is quantitively calculated by using a Gaussian kernel function with the spatial distance as reference. In addition, instead of utilizing the normal cross entropy loss in most existing methods during network training, we introduce an adaptive loss to handle class imbalance problem in MLS point clouds by dynamically adjusting the training weight of different categories according to their sample proportions in the training dataset.

2 The mechanism of geometry-aware attention module (GAAM)

To intuitively illustrate the mechanism of our designed GAAM in the feature space, we visualize and analyse the output feature maps of an input point set (4096×3) in three stacked GAAMs and MLP layers respectively. As shown in Figure 1, the learned features of each point marked in the red rectangle can be denoted by a feature vector (1×64) with 64 coloured squares and progressively abstracted into more significant features by three stacked encoding layers. Specifically, with the increase of feature encoding times, the point features learned by the simple MLP layer are suffered from serious feature loss and poor feature representation. Especially in the feature map generated by MLP3, most of input points with the values of feature channels tending to be zero fail to be encoded to obtain significant features. Comparatively speaking, as shown in the feature map generated by GAAM3, the learned point features are more informative and distinguishable. It demonstrates that the GAAM integrated by proposed GAANet can refine the vague feature maps by offsetting significant information loss and capturing more abstracted semantic features of point clouds.

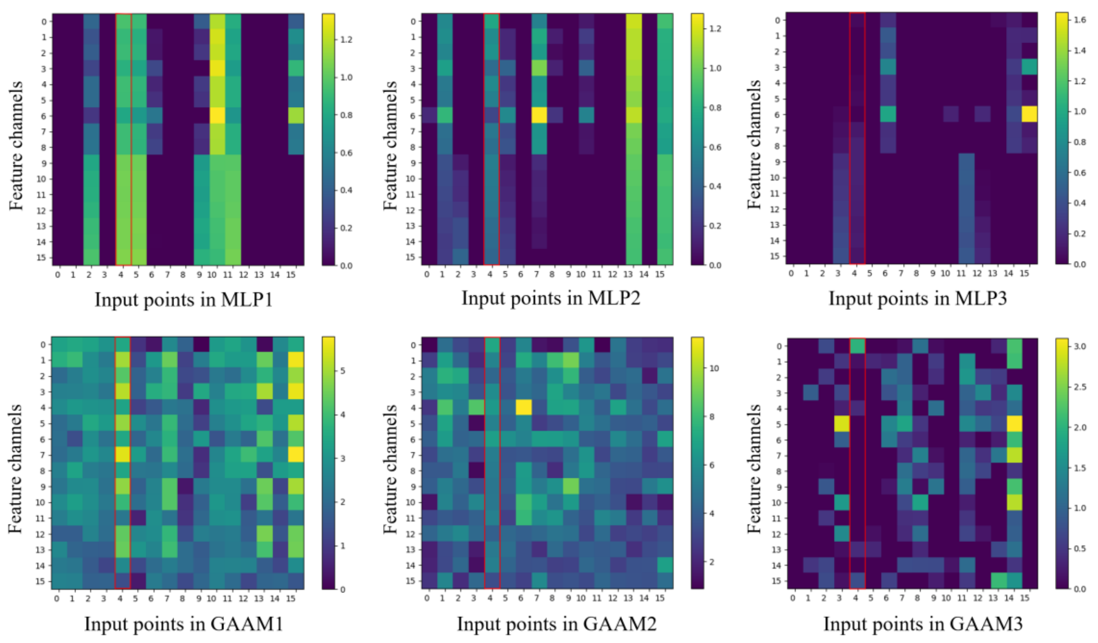


Figure 1. The visualization of the learned point features in different MLP layers and GAAMs. For better visualization, we only keep the part of top left corner of the whole output feature map (4096×64) as 16×16 image.

3 More experimental results and analysis

3.1 Loss function analysis

To validate the effectiveness of the loss function, we compare the proposed GAANet with four loss functions on the Oakland 3-D dataset and all experimental settings were consistent. The tested loss functions include cross entropy loss (L_{ce}), weighted cross entropy loss (L_{wce}), focal loss (L_{fl}) and the proposed adaptive loss ($L_{a\text{-}ce}$), respectively.

Table 1 shows the quantitative comparison of the proposed method with different loss functions. In Table 1, compared with the other three classical loss functions, the GAANet with our proposed $L_{a\text{-}ce}$ performs significantly better than the second place by 2.63% in mIoU, especially for some minority classes such as the pole, vegetation and wire. Obviously, the GAANet with L_{ce} gives the same training weight to different kind of point samples, which easily leads to local optimization on some majority classes and the reduction of overall performance. Although the GAANet with L_{fl} obtains almost the same score of OA as the GAANet with our $L_{a\text{-}ce}$, it is hard to find the suitable balance coefficient through experiments to achieve best overall performance.

Additionally, we also added the proposed $L_{a\text{-}ce}$ into other tested methods to verify its generalization ability. As shown in Table 2, all of the reference methods achieve substantial improvement in mIoU, which further proves the effectiveness of our proposed adaptive loss function.

Table 1. Comparison of the proposed method with different loss functions (%).

Methods	OA	mIoU	Facade	Ground	Pole	Vegetation	Wire
GAANet with L_{ce}	96.04	88.02	98.16	99.60	86.08	90.47	65.81
GAANet with L_{wce}	96.93	91.01	99.15	99.69	92.45	92.48	71.31
GAANet with L_{fl}	97.89	92.57	94.23	**99.77**	88.57	95.31	84.99
GAANet with $L_{a\text{-}ce}$ (Ours)	**98.69**	**95.20**	**99.37**	99.75	**90.93**	**96.95**	**86.47**

Table 2. Comparison of the proposed adaptive loss function used in different methods (%).

Methods	mIoU	Improvement
PointNet (Vanilla)	30.00	1.56
PointNet ++ (SSG)	47.33	**18.30**
DGCNN	78.34	11.19
GAPointNet	87.33	4.93
PointASNL	85.69	2.78

3.1 Computational requirements analysis

To illustrate the efficiency of the proposed method, we consider the computational requirements of several classical segmentation methods including PointNet, PointNet++, DGCNN, PointASNL and the latest GAPointNet by comparing the number of network parameters (Params), the floating point operations (FLOPs) which measures the complexity of network, and the inference time (s) of every 4096 points in the testing set of the Oakland 3-D dataset.

As shown in Table 3, although the PointNet has the lowest memory requirements with only 1.17M parameters and puts a lowest load on the processor of only 4.42×10^8 FLOPs, it yields poorest performance as presented in Section 4.3. By contrast, the proposed GAANet consumes modest computational and memory resources but achieves best overall performance.

Table 3. Computational resource requirements.

Methods	Params (million)	FLOPs (10^8)	Inference time (ms)
PointNet (Vanilla)	**1.17**	**4.42**	**5.95**
PointNet++ (SSG)	1.47	33.70	8.35
PointNet++ (MSG)	1.66	59.27	13.66
DGCNN	1.84	6.79	19.4
GAPointNet	2.30	9.67	33.9
PointASNL	2.24	47.78	58.0
GAANet (Ours)	1.90	7.39	31.6

3.2 Hyper-parameters analysis

To further investigate the impact of hyper-parameters set in our proposed method, the supplemental experiments are conducted with different hyper-parameter settings on the proposed network.

The experiments are first performed on the Oakland 3-D dataset with different numbers of nearest neighbours *k* in the local weighted graph and multi-heads *n* in each designed GAAM. Limited by the GPU memory, the possible *k* and *n* are not completely investigated. Table 4 shows the quantitative evaluation of the segmentation results. We can observer that the score of overall accuracy (OA) and mean IoU (mIoU) shows an increasing trend at first and then decreases with the increase of *k* and *n*. The proposed method obtains the highest score of 95.20 on mIoU and 98.69% on OA, respectively, when *k* is set to 20 and *n* is set to 4.

We note that a small number of neighbours (*k* of 10) and multi-heads (*n* of 2) are likely to make the network suffer the insufficient neighbouring feature embedding and poor feature learning, and excessive *k* or *n* values will likewise lead to a decrease in the performance due to the introduction of redundant point features and trap of local optimization during network training. Therefore, an appropriate setting of *k* and *n* has positive influence on the performance of the network.

Table 4. Effectiveness of different numbers of nearest neighbours and heads.

Neighbours	Heads	mIoU (%)	OA (%)
$k = 10$	$n = 2$	82.06	94.51
$k = 10$	$n = 4$	89.40	96.63
$k = 10$	$n = 8$	86.64	95.56
$k = 20$	$n = 2$	83.88	95.30
$k = 20$	$n = 4$	**95.20**	**98.69**

k = 20	*n* = 8	86.45	94.95
k = 40	*n* = 2	84.60	95.75
k = 40	*n* = 4	83.37	93.68

In addition, the experiments on the Toronto-3D dataset are then conducted with different balance coefficients γ of [0, 1, 2, 3, 5] in the proposed adaptive loss function. Seen from the Figure 2 (a), with the increase of γ value, the convergence speed of network becomes faster and the loss value reaches a lower level as the minority classes are assigned with the higher training weight. In Figure 2 (b), we can observe that the proposed GAANet achieves a relative lower score of 63.44% on mIoU as γ is set to 0, and the overall score presents a general trend of increasing at first and then decreases as the value of γ increases. Note that our GAANet obtains highest score of 65.09% on mIoU as γ is set to a modest value of 2 and the lowest score as γ is set to a higher value of 5, which means excessive higher or lower value of γ will be not conducive to balance the sample training of all classes and easily causes performance degradation. Just as illustrated in Figure 3, our GAANet shows a great diminishment on the performance of road and fence category when γ is set to a higher value of 5. By contrast, it shows a considerable performance on both the road and fence category when γ is set to 2.

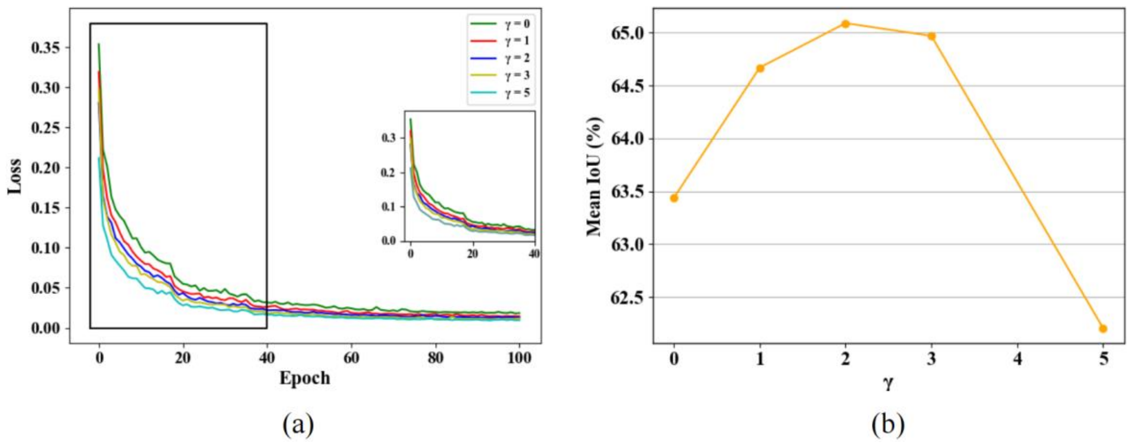
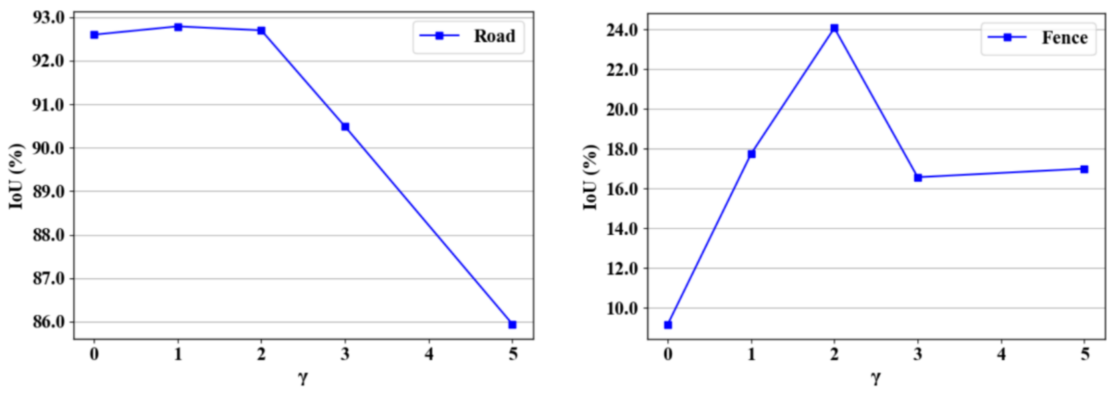


Figure 2. Performance of adaptive loss function with different balance coefficients. (a) The loss value curve on the training set; (b) The mIoU score curve on the testing set.



(a) (b)

Figure 3. The performance of the representative majority and minority class. (a) The IoU score curve of road (majority class) on the test samples; (b) The IoU score curve of fence (minority class) on the testing samples.